Applicazioni moderne: architetture di microservizi

Architettura moderna per l'agilità aziendale

Grazie alla diffusione di un'elaborazione veloce e di qualità, le aziende di tutte le dimensioni sono riuscite a creare efficienze interne e a raggiungere più clienti attraverso i prodotti digitali. Tuttavia, visti l'ubiquità degli strumenti, le opportunità di mercato multiple e il cambiamento nelle preferenze dei consumatori, per restare competitive le aziende devono innovare più in fretta che mai. In molti settori, da quello agricolo a quello bancario, passando per le telecomunicazioni, l'innovazione rapida è essenzialmente digitale. Il modo tradizionale di costruire i prodotti digitali non è abbastanza rapido da consentire l'innovazione necessaria per conquistare il mercato.

Nuovi pattern architetturali

I nuovi pattern architetturali come i microservizi consentono alle organizzazioni di accelerare il ritmo dell'innovazione. Grazie ad applicazioni moderne realizzate con le architetture di microservizi, l'innovazione deriva dalla distribuzione dell'impegno e dell'investimento tra team più piccoli e in un maggiore lasso di tempo, aumentando così la velocità di test e realizzando cambiamenti nel mercato. Ne consegue un'ottimizzazione dettagliata delle risorse e la possibilità per i team di dimensionare rapidamente i prodotti, sia per sviluppo sia per eseguibilità.

Quali sono le caratteristiche dell'architettura di microservizi?

Specializzazione

Ciascun servizio è progettato per un set di funzionalità e si concentra sulla risoluzione di un problema specifico. Se nel tempo gli sviluppatori apportano più codice a un servizio e questo diventa complesso, può essere suddiviso in servizi più piccoli.

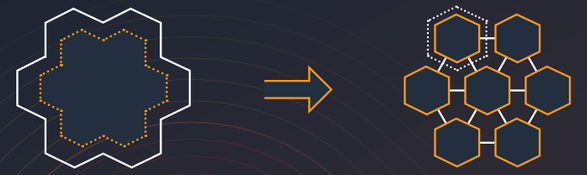
Distribuzione

L'architettura di microservizi fraziona il singolo processo di un'applicazione in componenti multipli che operano insieme per fornire valore. Le comunicazioni tra i singoli componenti avvengono mediante API con accoppiamento debole (loosely coupled) ben definite oppure tramite eventi e messaggistica.

Autonomia

In un'architettura di microservizi, è possibile sviluppare, distribuire, gestire e dimensionare ciascun servizio componente senza che ciò comprometta il funzionamento degli altri servizi. Non è necessario che i servizi condividano tra loro il codice o l'implementazione: si comportano come scatole nere a sé stanti.

Per quanto oggi un'architettura monolitica funzioni bene, la crescita dell'azienda può comportare delle sfide. Grazie ai microservizi, affronti le sfide comuni come il ridimensionamento e lo sviluppo di nuove funzioni rapidamente.



Scomposizione di un monolite

L'idea di scomporre un monolite può far paura. Fai fare prratica al tuo team invitandolo a prendere parte in un progetto AWS che lo accompagnerà nel processo di scomposizione di un monolite in microservizi. **Prova il tutorial >>**

Passaggio ai microservizi

Scopri come Mobvista ha adottato un'architettura di microservizi per migliorare la scalabilità e l'affidabilità della sua piattaforma. Leggi il caso di studio >>

Il pregio dei microservizi

Le architetture di microservizi sono realizzate con singoli elementi modulari che operano insieme. Anche se tale modularità non nasce per soddisfare l'aumento di superficie dell'area di codice, offre comunque vantaggi sostanziali in termini di innovazione più veloce, scalabilità indipendente, riduzione dell'impatto degli errori, sviluppo e distribuzione del codice.

I vantaggi dei microservizi

1. Agilità: un piccolo team che lavora su un singolo componente di servizio è liberato dai limiti degli altri componenti e pertanto sarà più rapido, rispondendo più velocemente a opportunità o problemi.

2. Facile distribuzione: i microservizi riducono le dimensioni dei cambiamenti, pertanto è semplice sperimentare nuove idee e tornare indietro se qualcosa non funziona.

3. Libertà tecnologica: con le architetture di microservizi, i team hanno la libertà di scegliere gli strumenti migliori per creare ogni singola parte dell'applicazione.

4. Scalabilità: grazie ai microservizi, è possibile dimensionare ciascun servizio in modo indipendente, così da soddisfare la domanda della singola funzionalità supportata.

5. Resilienza: i microservizi limitano l'impatto dell'errore a una singola parte dell'applicazione. Ciò significa che, in caso di errore di un singolo componente, viene diminuita la funzionalità pertinente senza provocare un crash dell'intera applicazione.

6. Codice riutilizzabile: dividere un software in componenti di servizi piccoli e ben definiti consente ai team di utilizzarli per scopi multipli all'interno della stessa applicazione. Ciò fa sì che un'applicazione si costruisca da sé, poiché gli sviluppatori possono creare nuove funzionalità sfruttando i servizi esistenti mediante le API senza dover scrivere il codice da zero o occuparsi dei dettagli dell'implementazione.

I microservizi per le applicazioni moderne

I microservizi aiutano l'organizzazione a migliorare la resilienza dell'applicazione e a ottimizzare la produttività del team. Il risultato è che i team di sviluppatori sono in grado di sperimentare e innovare più velocemente, così da rilasciare prodotti e funzionalità che offrono vantaggi competitivi.

Da dove iniziare

L'adozione di pattern architetturali dei microservizi non richiede un approccio "tutto o niente". I comuni percorsi verso un'architettura orientata ai servizi sono due: (a) incapsulare il monolite esistente in API trattandolo da scatola nera e nel frattempo costruire nuove funzionalità sotto forma di microservizi; (b) rifattorizzare il monolite in microservizi utilizzando il modello Strangler. Entrambe le strade hanno i loro pro e contro; qualunque percorso si scelga, ciò che è importante è anzitutto impostare l'infrastruttura di sviluppo adeguata. Ciò include la costruzione di pipeline di distribuzione del software automatizzate per costruire, testare e distribuire servizi eseguibili e infrastrutture in modo indipendente, al fine di mettere in sicurezza, monitorare e far funzionare un sistema distribuito ed eseguirne il debug.

Lasciare com'è il monolite è una soluzione che può funzionare se si tratta di un sistema standalone che non richiede aggiornamenti della funzionalità principale. In tal caso, la maggior parte dell'impegno per il nuovo sviluppo può essere rivolto alla costruzione di microservizi nuovi semplicemente connessi al monolite mediante API. Quando non è possibile mantenere né scartare il monolite ma alcune delle sue parti vanno riscritte, utilizzare il modello Strangler è l'approccio migliore.

Con il **modello Strangler**, i team di sviluppatori scompongono la funzionalità già parzialmente disaccoppiata dal monolite, per una facile sostituzione e disattivazione una volta costruito il microservizio. Ciò significa che i candidati ideali sono le capacità che non richiedono cambiamenti alle applicazioni destinate ai clienti e che potenzialmente non necessitano di un proprio archivio di dati. Ad esempio, per un'applicazione di e-commerce, i servizi potenziali da tenere in considerazione sono l'autenticazione, la fatturazione e i profili dei clienti. Al momento di costruire i loro primi microservizi molti team, anziché ottimizzare le funzionalità, mirano a testare e ottimizzare le pipeline di distribuzione del software e gli approcci API e a migliorare le competenze dei membri del team.

Visita il nostro sito per saperne di più sul potenziale dei microservizi per la tua azienda.